Latent Regression Forest: Structured Estimation of 3D Hand Poses

Danhang Tang, Hyung Jin Chang*, Alykhan Tejani and Tae-Kyun Kim

Abstract—In this paper we present the latent regression forest (LRF), a novel framework for real-time, 3D hand pose estimation from a single depth image. Prior discriminative methods often fall into two categories: holistic and patch-based. Holistic methods are efficient but less flexible due to their nearest neighbour nature. Patch-based methods can generalise to unseen samples by consider local appearance only. However, they are complex because each pixel need to be classified or regressed during testing. In contrast to these two baselines, our method can be considered as a structured coarse-to-fine search, starting from the centre of mass of a point cloud until locating all the skeletal joints. The searching process is guided by a learnt latent tree model which reflects the hierarchical topology of the hand. Our main contributions can be summarised as follows: (i) Learning the topology of the hand in an unsupervised, data-driven manner. (ii) A new forest-based, discriminative framework for structured search in images, as well as an error regression step to avoid error accumulation. (iii) A new multi-view hand pose dataset containing 180K annotated images from 10 different subjects. Our experiments on two datasets show that the LRF outperforms baselines and prior arts in both accuracy and efficiency.

Index Terms—random forest, regression forest, latent tree model, hand pose estimation, 3D, depth.

1 Introduction

Since the widespread success of real-time human body pose estimation [1], the area of hand pose estimation has received much attention within the computer vision community. Blooming starts when short-range depth sensors that are designed specifically for hand applications become available, including Primesense Carmine, Leap Motion and Intel RealSense. In this paper we introduce a novel method for real-time 3D hand pose estimation from a single depth image. To be more focussed, we limit the scope to only one hand, without interaction with the other hand or objects. Despite that, accurately and efficiently estimating the 3D pose of one hand is still beneficial to many human-computer interaction tasks. For instance, in the case of VR/AR, to enable interaction with virtual objects, 3D hand joint locations need to be recovered in real-time [2], [3], [4]. For some scenarios that are inconvenient for users to interact via other means, or hearing impaired people, gestures become a set-forward communication choice such as navigation [5], driver interaction [6]. In sign language recognition, not only 3D gestures, but the dynamics also plays a significant roles [7]. Recent works on in-air writing take a further step to free users from learning a sign language [8].

Judging by input of the classifier/regressor, traditional discriminative methods can be largely categorised into holistic and patch-based methods. Holistic methods take the whole hand region as input and predict the corresponding pose [9], [10]. They are efficient and always kinematically correct. However, due to the nearest neighbour nature, they usually do not generalise well to unseen samples.

Recent state-of-the-art body pose estimation techniques are patch-based, in which each pixel, considered as the centre of a local patch, is independently assigned a body part label [1] or a vote for joint locations [11], [12], [13]. Following this, several patch-based methods have been proposed for the 3D hand pose problem [14], [15]. In this case, predictors make decision for each pixel based on its local appearance. This allows combination between training data and thus can generalise to unseen samples to some degree. A downside of these methods is using only local information often produces results that break the anatomical constraints, i.e. , the 3D distances between joints, or the limited ranges of joint angles. Hence an ad-hoc global verification step is needed, as in [15]. Furthermore, the amount of input samples (foreground pixels) is usually hundreds of thousands, which not only increases the run-time complexity, but also exponentially enlarges the training set.

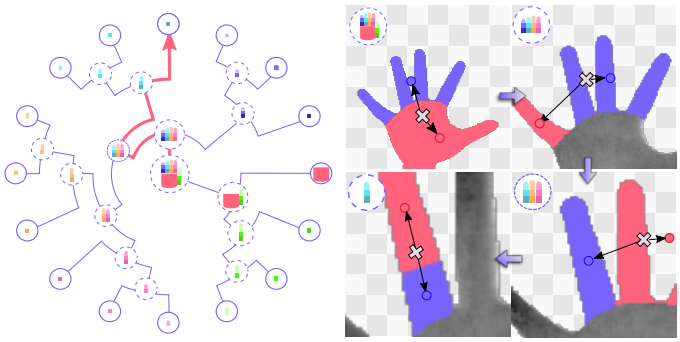


Fig. 1: Our method can be viewed as a coarse-to-fine search process for locating hand parts, guided by a binary latent tree model (LTM); starting from root of the LTM, we minimise the offset to its children at each level until reaching a leaf node which corresponds to a skeletal joint position. For simplicity, we only show the searching process for one joint. All figures are best viewed in colour and high-definition.

Despite the input size, state-of-the-art patch-based meth-

- *D. Tang is with perceptiveIO, Inc. Contact: see http://tang.danhang.net.*
- *H. Chang is the corresponding author.*
- *H. Chang and T-K Kim are with Imperial College London. Contact: see http://www.iis.ee.ic.ac.uk/ComputerVision*

ods can often run in real-time. The reasons behind are three-fold: (1) With depth data one can easily segment the foreground pixels - and yet there are still thousands. (2) Depth data provides scale information and thus no need to construct a scale pyramid as in RGB. (3) They usually adopt a very efficient but effective split function from [1]. Nonetheless, all these reasons have not yet considered the aforementioned differences between holistic and patch-based methods. Which means there is still room to further improve the runtime efficiency and accordingly the speed.

In this paper, we propose a novel algorithm, the *latent regression forest (LRF)* that combines the merits of both holistic and patch-based methods, i.e. runtime efficiency and generalisation. Specifically, we formulate the 3D hand pose estimation problem as a divide-and-conquer search for skeletal joints, guided by a learnt topological model of the hand. As shown in Fig. 1, our method starts by taking the whole hand as input, recursively dividing the input region into two parts that defined the topological model, until all skeletal parts are located. As the samples propagate down the LRF, the patch size shrinks from the whole hand to smaller local regions. This coarse-to-fine searching process allows combination between training samples. Since there is no need to harvest dense patches, it only requires a small training set for each stage, and maintains similar run-time complexity to holistic methods.

Furthermore, the hand topological model is used to enforce implicitly learnt global kinematic constraints on the output. Additionally, by training in a discriminative manner using our new diverse hand pose dataset, our approach is able to generalise to hands of various shapes and sizes as demonstrated in our experiments. Our experiments show that the LRF outperforms state-of-the-art methods in both accuracy and runtime efficiency. The main contributions of our work can be summarised as follows:

1) **Unsupervised learning of the hand topology:** We represent the topology of the hand by a latent tree model [16] which is learnt in an unsupervised fashion. This topological model is used for training the latent regression forest to enable a structured search.
2) **latent regression forest:** We introduce a framework for structured coarse-to-fine search in depth images. Guided by the learnt latent tree model, we learn regression trees that iteratively divide the input image into sub-regions until each sub-region corresponds to a single skeletal joint. Hence it is termed *Latent Regression Forests*.
3) **A new multi-view hand pose dataset:** We present a new hand pose dataset containing 180K fully 3D annotated depth images from 10 different subjects.

The previous conference version [17] has been extended by a more detailed discussion of the process of learning a latent tree model in Section 4.1, and the testing of latent regression forest in Section 4.2.2. More importantly, to better explain the idea, we introduce the holistic and patch-based baselines in Section 3 for comparison both conceptually and in experiments. In addition, we compare with a more up-to-date method [18] on another dataset MSHD in Section 5.5. More comparisons regarding runtime speed are reported in Section 5.6, as well as qualitative results in Section 5.7.

2 Related Work

In this section we will discuss the latest related work of the hand/body pose regression. For detailed surveys, we refer the reader to [19] for early work and [20] for recent work.

Many work for human body pose estimation take a discriminative approach by using large synthetic datasets in training and use either pixel-wise classification [1], or joint regression techniques [11], [12] for pose estimation. However, the hand has more self-occlusions and global viewpoints; thus these approaches require exponentially more data to capture this variation, making their direct application difficult. To address these, Keskin *et al* [14] propose a two-layered solution to the data-explosion problem. Due to the high degrees of freedom, capturing ground-truth annotated real data is a problem in its own right. Tang *et al* [15] investigate semi-supervised learning for hand pose estimation using annotated synthetic data and unlabelled real data. Tompson *et al* are the first who apply convolutional neural network (CNN) to the problem [21]. However, all these algorithms falls into the category of patch-based and hence can be potentially improved by our idea.

After the publication of our work [17], there has been a few methods taking the similar path of visual search. Li *et al* [22] take a similar coarse-to-fine approach to locate virtual landmark points. Alternatively, some methods perform a progressive search along a predefined structure, such as in human body [23], [24] and hand [25], [26] pose estimation.

As to generative methods, Oikonomidis *et al* [27] introduce a tracking based method for hand pose estimation in depth images using particle swarm optimisation (PSO). De la Gorce *et al* [28] incorporate shading and texture information into a tracker, whereas Ballan *et al* [29] use salient points on finger-tips for pose estimation. Melax *et al* [3] propose a tracker based on physical simulation which achieves state-of-the art performance in real-time.To avoid the complexity of rendering, Qian *et al* [30] combine PSO with an approximate sphere hand model.

A hybrid method combines discriminative and generative concepts [18], [31], [32]. Similarily [21], [33] also fall into the hybrid category but rather than exploiting the rendered hypotheses they use PSO to find the best location hypothesis provided by the discriminative part. Krejov *et al* [34] uses forest as initialisation but the optimisation part follows a physics-based process as in [3]. Oberweger *et al* using only CNN to imitate a generative loop [35]. An interesting method proposed by Choi *et al* [36] uniquely formulates the problem as a cold-start problem in recommender systems. Our method, being only an efficient discriminative solution, can be combined with generative methods too.

Tree-based graphical models have recently been used for estimating human body pose. Tian *et al* [37] build a hierarchical tree models to examine spatial relationships between body parts. Whereas Wang and Li [38] use a latent tree model to approximate the joint distributions of body part locations. However, both methods do not exploit the hierarchy as a coarse-to-fine search as our method, hence for each part detector, all possible locations need to be scanned. Furthermore, recent methods for constructing these models [16], [39] enable us to learn consistent and minimal latent tree models in a computationally efficient manner.

3 BASELINES

We decompose the hand pose estimation problem into estimating the location of 16 skeletal parts on the hand model, as illustrated in Fig. 2. More formally, we are given a depth image I of a segmented hand and trying to predict a set of 3D positions for 16 skeletal parts $\mathcal{P}$. To this end, a training set $\mathcal{S} = \{(I, \mathcal{P})\}$, where each training sample can be represented as a tuple $(I, \mathcal{P})$ is used to train a classifier. In this paper, we choose the decision forest [40], [41] as a basic predictor component because: 1) decision forest inherently supports multiple classes which lies in the nature of the hand pose problem, 2) decision forest is usually very efficient and has achieved real-time performance in numerous vision problems, 3) it has been successfully applied to body pose [1] and hand pose [15]. However, we note that our method can adopt other machine learning components.

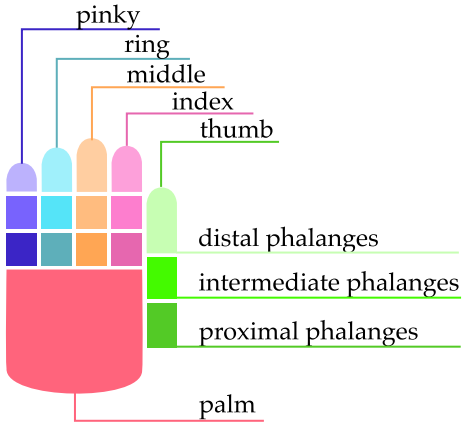


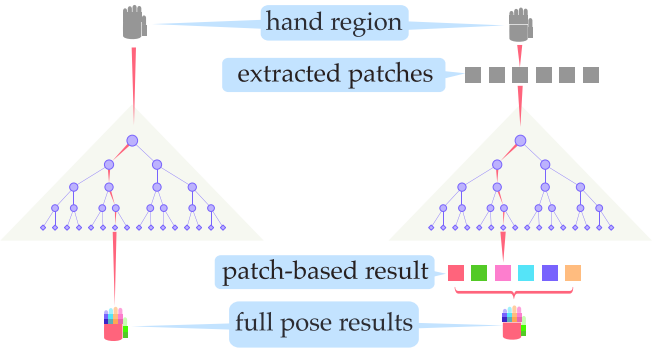
Fig. 2: Colour coding of the 16 skeletal parts in this paper.

Conceptually a decision forest is an ensemble of decision trees, whereby at test time each internal node routes data to its left or right childnode by applying a threshold to a projection of the input features. Each input ultimately ends up at a leaf node, where a prediction function is stored during training and applied during testing. A regression forest differs from a classification forest in two aspects: 1) the training quality function is designed to minimise the intra-childnode spatial variance; 2) the leaf models predict continuous output values such as hand part locations.

In this section, we describe a holistic regression forest and a patch-based regression forest, respectively in Section 3.1 and 3.2. They will serve as baselines in the experiment part (see Section 5).

3.1 A holistic method

We now design a simple holistic regression forest, which takes the entire hand region as input, and regresses directly on 16 hand part locations. To this end, we define the input as a tuple (I, ρ_0), where ρ_0 is the centroid of segmented pointcloud, and the output as $\vec{\theta} = ||(\rho_k - \rho_0 | \rho_k \in \mathcal{P})$, where each part location is normalised with the centroid and hence becomes an offset, and $||$ is the concatenation operator resulting $\vec{\theta} \in \mathbb{R}^{48}$ (concatenating 16 3D offset vectors). The training data for this holistic regression forest hence becomes $\mathcal{S}_H = \{(I, \rho_0, \vec{\theta})\}$.



(a) A holistic method (b) A patch-based method

Fig. 3: Comparing holistic and patch-based regression trees. (a) a holistic regression tree takes the whole hand as input and gives one pose result; (b) a patch-based regression tree takes each patch as input and aggregates results.

During training, for each internal node, the following quality function is used to find a sub-optimal split,

$$Q(\mathcal{S}) = \mathrm{tr}(\Sigma^{\mathcal{S}}) - \sum_{k}^{\{l,r\}} \frac{|\mathcal{S}^k|}{|\mathcal{S}|} \left(\mathrm{tr}\left(\Sigma^{\mathcal{S}^k} \right) \right), \qquad (1)$$

where $\Sigma^{\mathcal{X}}$ is the covariance matrix of the set of offset vectors $\vec{\theta}$ and $\mathrm{tr}(\cdot)$ is the trace function. Note that Eq. (1) need to minimise the variance of 48D vectors, which complexity will be reduced by our method.

We adopt the feature function from [1], [15].

$$f(\boldsymbol{\rho}, I) = I\left(\boldsymbol{\rho}^I + \frac{\mathbf{u}}{I(\boldsymbol{\rho}^I)}\right) - I\left(\boldsymbol{\rho}^I + \frac{\mathbf{v}}{I(\boldsymbol{\rho}^I)}\right), \qquad (2)$$

where $\boldsymbol{\rho}^I$ is the projection of $\boldsymbol{\rho}$ on image space (2D).

3.2 A patch-based method

We introduce a patch-based baseline which is based on the work of Keskin *et al* [14]. It has a two-layer structure: first classify hand images into different shapes and then classify each pixel (or patch) into one of the hand parts. Such a structure saves the memory and can usually improve the accuracy. However, this classification-based method cannot predict self-occluded parts. We then modify it by integrating regression voting as in [15].

In this case, each input sample is defined as (I, ρ_x), where ρ_x is the absolute location of a foreground pixel x. The corresponding output is $\vec{\theta}_x = ||(\rho_k - \rho_x | \rho_k \in \mathcal{P})$, $\vec{\theta}_x \in \mathbb{R}^{48}$. Note that each pixel x has its own offset $\vec{\theta}_x$, whereas in the case of holistic there is only one offset $\vec{\theta}$ to the hand centroid. The training data is then given as $\mathcal{S}_P = \{(I, \rho_x, \vec{\theta}_x)\}$.

The same quality Eq. (1) and feature Eq. (2) functions can be adopted here. However, since $|\mathcal{S}_H| \ll |\mathcal{S}_P|$ due to the size of $\{x\}$, it is much more efficient to train a holistic regressor.

The results from all patches and trees are then aggregated together and a robust mean-seeking method, in our case *Meanshift* [42], is applied to locate the final hand part positions. The differences between a holistic regression tree and a patch-based regression tree are illustrated in Fig. 3.

4 Proposed Method

Our method can be formulated as a divide-and-conquer search, in which the input image is recursively divided into two cohesive sub-regions, until each sub-region contains only one skeletal part. To attain robustness to the complex articulations of human hand, the search is carried out in a structured, coarse-to-fine manner, where the granularity of each search stage is defined by a learnt topological model of the hand.

In Section 4.1 we discuss how we can learn the hand topology in an unsupervised fashion. Following this, in Section 4.2, we discuss how this topology is used to build a latent regression forest (LRF) to perform a structured, coarse-to-fine search in the image space.

4.1 Learning the hand topology

To guide the search process, we desire to define a coarse-to-fine, hierarchical topology of the hand, where the coarsest level of the hierarchy is given by the input to the search, i.e. , the entire hand, and the finest level by the outputs, i.e. , the skeletal parts. Using a latent tree model (LTM) to represent this structure is a sensible choice. More importantly, unlike previous works that use LTM [38], our method aims to leverage this hierarchical model to reduce the training samples and testing complexity.

Wang and Li applied LTM to represent the articulation of human body [38]. However, their method is not an ideal choice for our problem because: 1) The default LTM they learnt contains no latent node, i.e. , all nodes represent skeletal parts. 2) They then defined 10 combined parts (or poselets) to mimic latent nodes, resulting in an LTM that does not seem to be the most representative topology of human body. Both cases do not support our coarse-to-fine search paradigm and hence do not improve the efficiency.

In this section, we introduce a method to automatically learn an LTM that not only captures the hand topology, but also is a coarse-to-fine manner. This method requires neither prior knowledge of physical joint connections, nor predefined combined parts, which means it can be applied to any other articulated objects.

4.1.1 Model definition

An LTM is a tree-structured graphical model, $\mathcal{M} = (O \cup L, E)$, where the vertices are composed of *observable vertices* (skeletal parts), O, and *latent vertices* (combination of parts), $L = \{l\}$, where $l \subseteq O$; and E denotes edges (see Fig. 6 (c) (d)).

Given an LTM of the hand topology, $\mathcal{M}$, for each vertex $i \in \mathcal{M}$, $i = 0...|\mathcal{M} - 1|$, its parent is defined by $p(i)$ and its 2 children by $l(i)$ and $r(i)$. For each training depth image, I, a 3D position, $\boldsymbol{\rho}_i^I$, is associated with i. For each observable vertex, $i \in O$, this is simply the position of the associated skeletal part; for each latent vertex, $i \in L$, the position is represented by the mean position of the observable nodes they are composed of.

4.1.2 Unsupervised learning

A strategy called Chow-Liu neighbour joining (CLNJ) for learning LTMs efficiently was proposed by Choi *et al* [16]. The method starts by constructing an information distance matrix $\mathbf{D}$ of all random variables. Then a Chow-Liu tree is constructed with $\mathbf{D}$. For each internal node in the Chow-Liu tree, a recursive joining method scheme is applied by identifying its neighbourhood. This method can produce consistent LTMs without redundant latent nodes. Please refer to [16] for more details.

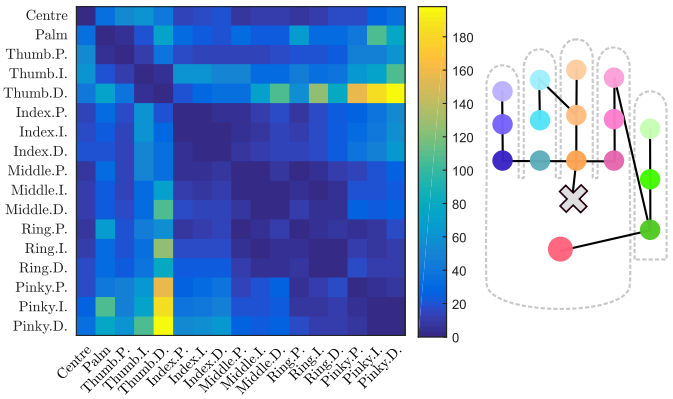


Fig. 4: The distance matrix (left) and latent tree model (right) generated with the information distances defined in [16].The cross symbol indicates the centroid. (P: proximal phalanx, I: intermediate phalanx, D: distal phalanx, the same in other figures.)

Recall that we are given a training set $\mathcal{S} = \{(I, \mathcal{P})\}$, where $\mathcal{P}$ which represents 16 skeletal part positions can now be related to observable vertices, such that $\mathcal{P} = \{\rho_i | i \in O\}$. Therefore we do not assume any knowledge of physical part connections and do not define any combined parts, which our algorithm will adaptively learn.

At the beginning of the search, we have no knowledge of any joint location but a region of interest containing a segmented pointcloud. Hence in addition to the 16 part locations, we define the holistic hand region as one more variable and the centroid of the segmented pointcloud as its location. Note that this should be considered as an observable node rather than a latent node, since its location is calculated from the pointcloud rather than other part positions. This change simply provides a 'starting point' for our algorithm.

4.1.3 Distance function

A distance matrix, $\mathbf{D}$, of all 17 vertices (1 centroid and 16 parts) is needed for CLNJ. The key part of constructing $\mathbf{D}$ is to define a distance function. In [16] the *information distance* between two random variables x and y is defined as below,

$$d_{xy} = -\log \left| \frac{\mathrm{cov}(x, y)}{\sqrt{\mathrm{var}(x)\,\mathrm{var}(y)}} \right| \tag{3}$$

This results in a distance matrix and an LTM as shown in Fig. 4, which are consistent with the LTM in [38], i.e. , no latent nodes. As mentioned before, we desire to learn a graphical model that represents a coarse-to-fine structure, where predictions of each level become a step closer to the final results. Hence we explore two other different metrics that better represent the physical connections.

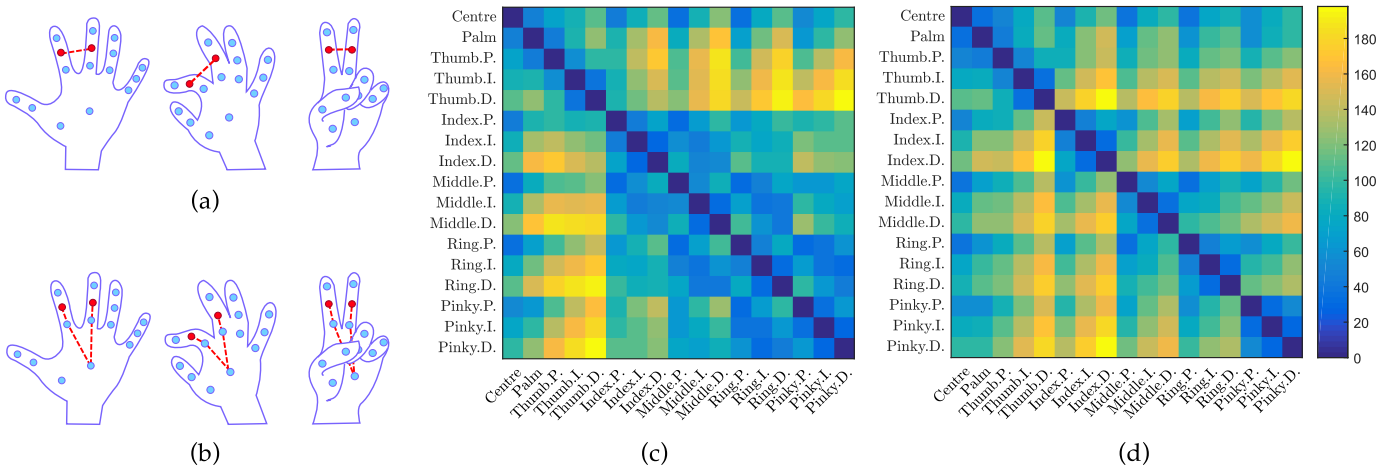


Fig. 5: Comparison between using Euclidean and geodesic distance: (a) Euclidean distance between two joints, (b) geodesic distance between two joints, (c) distance matrix generated with Euclidean distance, (d) distance matrix generated with geodesic distance.

Using the training set $\mathcal{S}$, we define the distance d between two observable vertices x and y as:

$$d_{xy} = \frac{\sum_{I \in \mathcal{S}} \delta\left(I, x, y\right)}{|\mathcal{S}|}, \tag{4}$$

where $\delta\left(I, x, y\right)$ is a function measuring the distance between vertices x and y in image I. An intuitive choice for δ is Euclidean distance. However, we find that the learnt LTM cannot represent the anatomical structure well (see Fig. 6). To better exploit the training data, we propose to use a geodesic-like function for measuring pair-wise part distance.

Formally, the process of calculating the geodesic distances of a sample $(I, \mathcal{P})$ is as follows,

1) We construct a fully connected, undirected graph of all 17 vertices in $\mathcal{P}$, using Euclidean distance.
2) Then for each pair of vertices x, y in this graph, a projected 2D Bresenham line is drawn on I and all the depth values along this line are averaged.
3) If the averaged depth value is larger than $\max(I(x), I(y))$, the edge between x and y is removed. This means there is a large depth discontinuity along the edge in image space, i.e. , the Bresenham line passes both foreground and background.
4) What remains is a graph in which the edges all lie along a smoothly transitioning depth path. The geodesic distance between two vertices can then be calculated as the shortest path in this graph.

In Fig. 5 (a) and (b) we demonstrate the difference between Euclidean and geodesic distance on a toy hand model. Through different poses the Euclidean distance between two joints can change drastically while the geodesic one remains largely unchanged; this robustness to pose is preferable to the capture the topology of human hand. This is also reflected in the distance matrices shown in Fig. 5 (c) and (d). The distances between parts from different fingers are larger using geodesic distance. Note that the geodesic distance still fails in some poses when there is no depth discontinuity, e.g. , the fist pose. Since Eq. (4) takes all training samples into account, the final $\mathbf{D}$ will still reflect the desired kinematic structure.

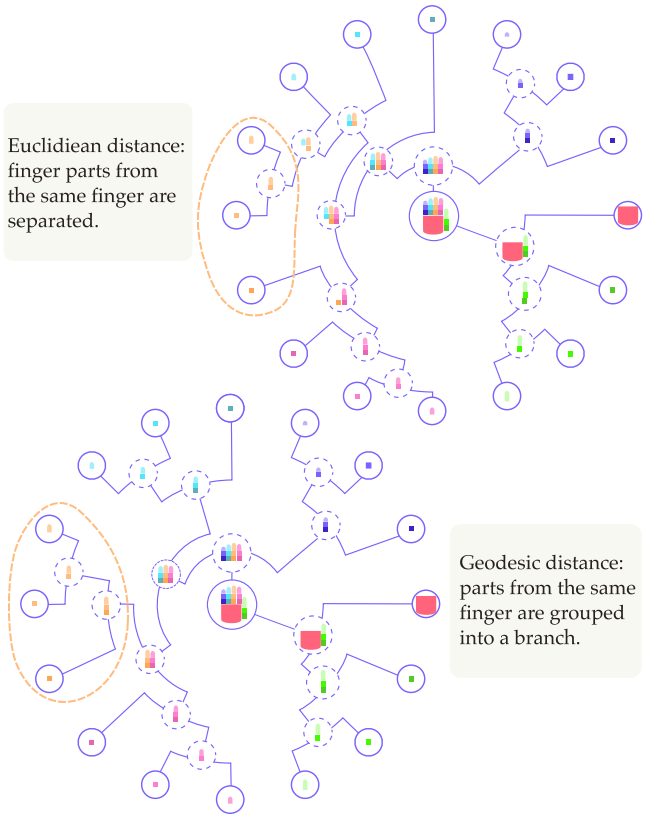


Fig. 6: Learnt latent tree models. Top: using Euclidean distance. Bottom: using geodesic distance. Solid circles represent observable vertices and dashed ones latent vertices. Yellow dash lines encircle the three parts of middle finger. They are separated In the case of Euclidean distance and grouped together in the case of geodesic distance.

In Fig. 6, we illustrate two LTMs generated using the Euclidean and geodesic metrics respectively. There are two interesting observations comparing to Fig. 4: 1) A coarse-to-fine structure is represented with latent nodes. 2) The model is a binary tree - this is relatively trivial, since it

is an outcome of the distance function and our method does not have this constraint. As highlighted by the dashed lines, the Euclidean-generated model groups sub-parts of fingers with different fingers e.g. proximal phalanx of middle finger is grouped with index finger, whereas in the geodesic-generated model the fingers are all separated. This is consistent to the anatomical structure, which we assume to have no knowledge during training. This means our method can be applied to any other articulated objects for recovering topology and pose estimation. These two different structures results in the different structure of our regressor, which is covered in Section 4.2. The impact on accuracy is empirically demonstrated in Section 5.

4.2 Latent Regression Forest

The aim of a latent regression forest (LRF) is to perform a search of an input image for several sub-regions, each corresponding to a particular skeletal part. Searching is preformed in a divide-and-conquer fashion where each division is guided by the learnt latent tree model representing the topology of the hand.

An LRF is an ensemble of randomised binary decision trees, each trained on a bootstrap sample of the original training data. Each latent regression tree contains three types of nodes: *split*, *division* and *leaf* (see Fig. 7). Split nodes perform a test function on input data and decides to route them either left or right. Division nodes divide the current search objective into two disjoint objectives and propagate input data down both paths in parallel. Finally, leaf nodes are terminating nodes representing a single skeletal part and store votes for the location of this joint in 3D space.

In Section 4.2.1 we discuss how to build the LRF followed by a discussion of the testing procedure in Section 4.2.2.

4.2.1 Training

Each latent regression tree (LRT) in the LRF is trained with the aforementioned dataset $\mathcal{S}$: the LRT is trained in stages, where each stage corresponds to a non-leaf vertex in the LTM, $\mathcal{M}$. Starting with the root vertex, $i = 0$, of $\mathcal{M}$ we grow the LRT with the objective of separating the image into two cohesive sub-regions which correspond to the vertices, $l(i)$ and $r(i)$, which are the children of the root node.

This separation is achieved by growing a few layers of latent regression tree. At each node, we randomly generate splitting candidates, $\Phi = \{(f_i, \tau_i)\}$, consisting of a function, f_i, and threshold, τ_i, which splits the input data, $\mathcal{S}$, into two subsets, $\mathcal{S}^l$ & $\mathcal{S}^r$, s.t. $\mathcal{S}^l = \{I | f_i(I) < \tau_i\}$ and $\mathcal{S}^r = \mathcal{S} \setminus \mathcal{S}^l$. A function, f_i, for a splitting candidate, whilst at the stage represented by the LTM vertex i is defined as:

$$f_i(\boldsymbol{\rho}_i^I, I) = I\left(\boldsymbol{\rho}_i^I + \frac{\mathbf{u}}{I(\boldsymbol{\rho}_0^I)}\right) - I\left(\boldsymbol{\rho}_i^I + \frac{\mathbf{v}}{I(\boldsymbol{\rho}_0^I)}\right), \quad (5)$$

where $I(\cdot)$ is the depth at an image position, $\boldsymbol{\rho}_i^I$ is the position of the LTM vertex, i, in the image, I and vectors $\mathbf{u}$ and $\mathbf{v}$ are random offsets. Similarly to Eq. (2), the offsets are normalised to make them depth-invariant. However, in order to avoid error accumulation in depth values the normalisation factor is always set to the centroid, $\frac{1}{I(\boldsymbol{\rho}_0^I)}$. This is because the position of a latent node may not be on the foreground point cloud. It is however guaranteed that the depth value of centroid is valid.

Algorithm 1 Growing a latent regression tree

Input: A set of training samples $\mathcal{S}$; a pre-learned LTM $\mathcal{M} = (O \cup L, E)$ with maximum depth D.
Output: An LRF T

procedure GROW($\mathcal{S}$, M)
  Equally divide $\mathcal{S}$ into random subsets $\mathcal{S}_0, ...\mathcal{S}_D$
  Let $i = 0$, $j = 0$ ▷ Initialise ith node of LTM and jth node of LRF
  Let $d = 0$ ▷ First stage of training
  SPLIT($i, j, \mathcal{S}_0, d$)

function SPLIT(i, j, $\mathcal{S}$, d)
  Randomly propose a set of split candidates Φ.
  for all $\phi \in \Phi$ **do**
    Partition $\mathcal{S}$ into $\mathcal{S}^l$ and S^r by ϕ with Eq. (5).
  Use the entropy in Eq. (6) to find the optimal $\phi*$
  if $IG_i(\mathcal{S})$ is sufficient **then**
    Save j as a *split node* into T.
    SPLIT(i, $l(j)$, $\mathcal{S}^l$, d)
    SPLIT(i, $r(j)$, $\mathcal{S}^r$, d)
  else if $i \in L$ **then**
    Save j as a *division node* into T
    Let $\mathcal{S} = \mathcal{S} \cup \mathcal{S}_{d+1}$
    SPLIT($l(i)$, $l(j)$, $\mathcal{S}$, $d + 1$)
    SPLIT($r(i)$, $r(j)$, $\mathcal{S}$, $d + 1$)
  else
    Save j as a *leaf node* into T.
  Return

The splitting candidate, ϕ_i^*, that gives the largest information gain is stored at the LRT node, which is a *split node* as in the standard Random Forest. The information gain whilst at the stage represented by the LTM vertex i is defined as:

$$Q_i(\mathcal{S}) = \sum_{m}^{l(i),r(i)} \mathrm{tr}(\Sigma_{im}^{\mathcal{S}}) - \sum_{k}^{\{l,r\}} \frac{|\mathcal{S}^k|}{|\mathcal{S}|} \left(\sum_{m}^{l(i),r(i)} \mathrm{tr}\left(\Sigma_{im}^{\mathcal{S}^k}\right) \right) \quad (6)$$

where $\Sigma_{im}^{\mathcal{X}}$ is the sample covariance matrix of the set of offset vectors $\left\{ \left(\boldsymbol{\rho}_m^I - \boldsymbol{\rho}_i^I\right) | I \in \mathcal{X} \right\}$ and $\mathrm{tr}(\cdot)$ is the trace function. The offset vectors indicate the offsets from the current centre to each centre of the two subregions. Even when compared to the holistic baseline, since the offset vectors are 6D, this is more efficient than Eq. (1) (48D).

This process is then repeated recursively on each split of the data, $\mathcal{S}^l$ & $\mathcal{S}^r$, until the information gain falls below a threshold.

At this point we introduce a *division node* which divides the current search objective into two finer ones and enters the next search stage. The division node duplicates the training data and continues to grow the tree along two separate paths, each corresponding to to one of the two children of the current LTM vertex, i. Additionally, for each training image, I, reaching this division node we store the vectors $\boldsymbol{\theta}_m = \left(\boldsymbol{\rho}_m^I - \boldsymbol{\rho}_i^I\right)$ corresponding to the 3D offsets of i and its children $m \in \{l(i), r(i)\}$.

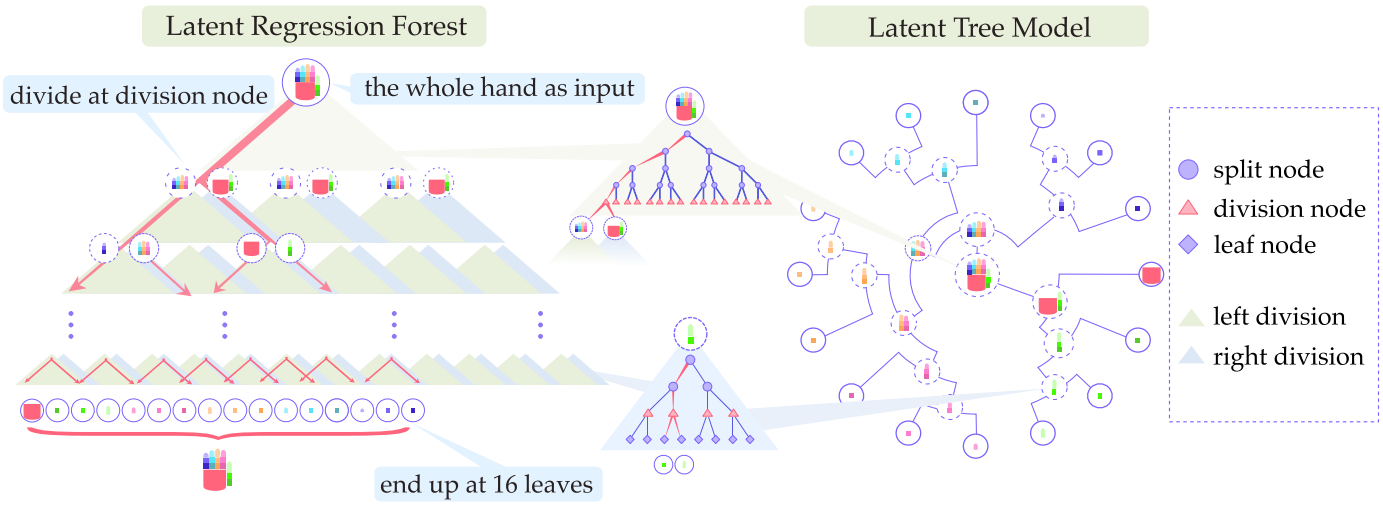


Fig. 7: An comparison between (a) a traditional Regression Tree, where each patch sample is extracted from a depth image, propagated down the tree and ends up at one leaf node; and (b) a latent regression tree, where the whole point cloud is propagated down the tree and keep dividing until ending up at 16 leaf nodes.

This process of split followed by division is then repeated until the LTM vertex, i, to be considered is a leaf; at which point we create a leaf node in the LRT corresponding to the skeletal part represented by i. The leaf node stores information about the 3D offset of i from its parent $p(i)$, that being $\left(\boldsymbol{\rho}_i^I - \boldsymbol{\rho}_{p(i)}^I\right)$.

Since the hand has many complex articulations and self-occlusions, in order to fully capture this variation the training set used is extremely large. To retain training efficiency we make use of the fact that we train in coarse-to-fine stages based on the learnt LTM. An intuition is that coarse stages require less training data than the fine ones, therefore we can gradually add more training data at each stage of the training procedure.

For an LTM of maximum depth D, we split the training data, $\mathcal{S}$, into D equally sized random, disjoint subsets $\mathcal{S}_0, ...\mathcal{S}_{D-1}$. We start training an LRT with $\mathcal{S}_0$ for the first stage, and for each stage after we add an additional subset to the training data. That is, for stage d the training set is composed of $\mathcal{S}_d \cup \mathcal{S}_{d-1}$. The training procedure to grow a single LRT is described in Algorithm 1.

Although a multi-stage coarse-to-fine structured search is efficient, an underlying risk is that the dependency between stages can lead to error accumulation throughout the search. To compensate for this, we embed an *error regressor* inspired by [43] into each stage of latent regression forest. After training stage d with set $\mathcal{S}_d$ and before creating a *division node* , we use S_{d+1} to validate the trained forest so far. For each sample $s_i \in \mathcal{S}_{d+1}$, an error offset $\Delta\boldsymbol{\theta}$ between the ground truth and the estimation is measured. Similar to the previously described method of splitting, the forest is further grown for a few layers in order to minimise the variance of $\Delta\boldsymbol{\theta}$. Once the information gain falls below a threshold a *division node* is generated and the forest training enters next stage, $d+1$.

4.2.2 Testing

At test time, pose estimation is performed on an image I as follows; we define the starting position for the search, $\rho_{i=0}^I$ as the centre of mass of the depth image, which corresponds to the root vertex of the LTM. Starting at the root of the latent regression forest, the image traverses the tree, branching left or right according to the split-node function, until reaching a division node. For each offset, $\boldsymbol{\theta}_j$ stored at the division node, 3D votes are accumulated in two Hough spaces, $\mathbb{H}^l$ and $\mathbb{H}^r$, where the votes for $\mathbb{H}^l$ are defined as $\left\{\rho_i^I + \frac{\boldsymbol{\theta}_j}{\rho_0^I} | \boldsymbol{\theta}_j \in \boldsymbol{\theta}_l\right\}$ and similarly for H^r. The modes of these two Hough spaces now represent the two new positions, $\rho_{l(i)}^I$ and $\rho_{r(i)}^I$, from which the next search stage begins. This process is then repeated recursively until each path terminates at a leaf node.

This process will result in the image reaching *multiple leaf nodes*, one for each terminating node in the LTM. Using the stored offsets at the leaf nodes, each leaf node votes for its corresponding skeletal part in a corresponding 3D Hough space. Aggregating votes of all trees, we locate the final positions of the joints by a structured search in the Hough space, for which the structure is dictated by the learnt LTM as follows. For each skeletal part, we assign to it a dependent observable vertex in the LTM which corresponds to the vertex with the smallest geodesic distance as calculated in the matrix, $\mathbf{D}$ (Eq. (4)). The location of each joint in the Hough space is then defined as the maximum which is closest to the location of its dependent vertex.

4.2.3 Performance Issues

In contrast to the patch-based baseline that takes dense pixels as input [14] our algorithm takes the entire hand region. Thus, while both methods are constrained in complexity by the depth of the trees d, ours processes much smaller amount of samples. This is because the number of pixels to be evaluated in patch-based approaches are usually in the order of thousands for a standard VGA image; whereas, in contrast, we only evaluate one sample per image. The complexity is then similar to the holistic baseline - though a bit higher due to the division. Since there are too many different factors, it is hard to compare the complexity formally. Instead we report the run-time speed in Section 5.

On the other hand, recall that the training of holistic and patch-based baselines require to minimise the variance of 48 dimension offset vectors. In LRF, since the offset vectors are 6D, it is much faster for training one split node.

Algorithm 2 Testing with LRF

Input: A segmented depth image I; a forest $\mathcal{F}$; an LTM $\mathcal{M}$
Output:

procedure TEST(I)
  Calculate centroid ρ_0 of I.
  for all $t \in \mathcal{F}$ **do**
    Let $i = \mathcal{M} \rightarrow root$.
    Let $j = t \rightarrow root$.
    PROPAGATE(ρ_0, i, j)
  Aggregate results from all trees using *Meanshift*.

function PROPAGATE(ρ, i, j)
  if j is a *leaf node* **then** ▷ *leaf node*
    Vote for the hand parts indicated by i.
  else if j is a *division node* **then** ▷ *division node*
    Predict centres of two subregions (ρ_l, ρ_r) with stored votes in j.
    PROPAGATE($\rho_l, i \rightarrow l, j$) ▷ left division branch.
    PROPAGATE($\rho_r, i \rightarrow r, j$) ▷ right division branch.
  else ▷ *split node*
    if $f_i(\boldsymbol{\rho}_i^I, I) == 1$ **then**
      PROPAGATE($\rho_l, i, j \rightarrow l$)
    else
      PROPAGATE($\rho_r, i, j \rightarrow r$)

5 EXPERIMENTS

5.1 Datasets

For experiments, we collect a new dataset called the Imperial College Vision Lab (ICVL) dataset. The Intel®'s *Creative Interactive Gesture Camera* [3] is used for capturing training and testing data. As the state-of-the-art consumer time-of-flight sensor, it captures depth images at a lower noise level than structured-light sensors making them ideal for hand pose estimation. For labelling, we utilise [3] to obtain a preliminary pose for each frame, and then manually refine.

For training, we have collected sequences from 10 different subjects with varying hand sizes by asking each subject to make various hand poses with an illustration of 26 different postures shown as aid. Each sequence was then sampled at 3fps producing a total of 20K images and by additionally applying in-plane rotations to this set, the final dataset contains 180K ground truth annotated training images. For testing, we have collected two sequences (denoted sequence A and B) each containing 1000 frames capturing a vast array of different poses with severe scale and viewpoint changes. Furthermore, as [3] is tracking based and requires initialisation (frontal view of an open hand), in order to do a fair comparison both test sequences start in this way. Example images from our ICVL dataset are shown in Fig. 8.

Since the used annotation tool often fails at difficult poses, the ICVL dataset is dominated by frontal poses. As a complement, we have also experimented on the Microsoft Research Synthetic Hand Dataset (MSHD) [18], which was

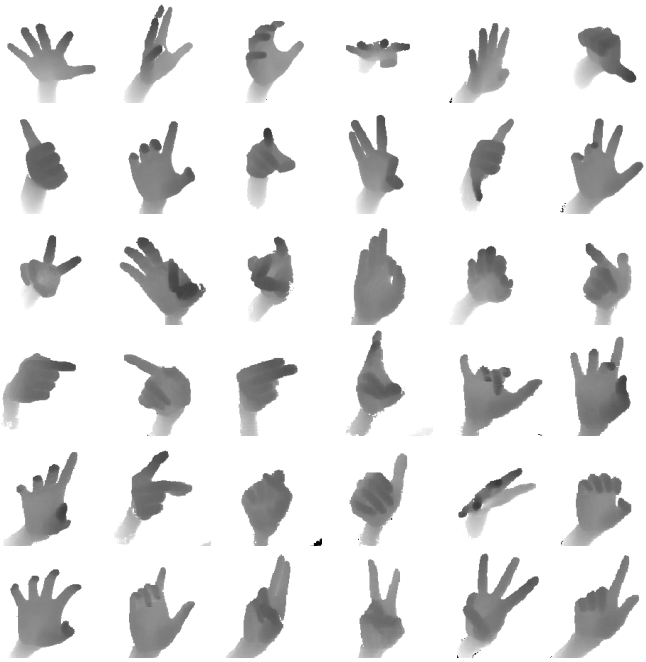


Fig. 8: Examples from our ICVL dataset. These images are in-plane rotated to cover the *roll* degree. Depth images are segmented, cropped and normalised for visualisation.

synthetically generated with camera intrinsic of Kinect2. Training set has 100k images whilst the testing set has 1k images. They provide pose parameters in 3D Euclidean and angular forms, as well as pixel-wise classification labels. Despite both training and testing data are synthetic, MSHD is very challenging because of the following reasons. 1) It is the first hand dataset that uniformly covers the full viewpoint space. The amount of training data is relatively sparse considering the covered parameter space, which makes a learning-based method rather difficult to achieve low error on this dataset. 2) When generating, it takes shape and size into consideration by randomly choosing one of the 13 personalised mesh models. These models have been fitted to different subjects using [44]. 3) All testing images are randomly generated without temporal information, which makes it very difficult for any method that relies on tracking. 4) Although being synthetic, the rendering process explicitly simulates the noise of time-of-flight sensors.

5.2 Setup

In all experiments we train each latent regression tree by evaluating 2000 splitting candidates at each node and the threshold used to stop growing the tree at a particular stage is chosen based on the size of a finger joint, which was set to $(10mm)^2$. Apart from the experiment for deciding number of trees, we train a 16-tree LRF for the other experiments. To be fair in comparison, the same amount of trees and stop condition are also used for the baselines.

In Section 5.3 we conduct a self comparison of the different components in the latent regression forest. Following this, we perform a thorough evaluation with state-of-the-art methods on the ICVL dataset Section 5.4 and MSHD

dataset Section 5.5. Efficiency is reported in Section 5.6 Finally, in Section 5.7 we present some qualitative results.

5.3 Parameter Choices

To evaluate the impact of different distance metrics used when constructing the LTM, we quantitatively measure the impact of the different topologies on performance. All experiments within this section are conducted on the ICVL dataset. We compare LTMs generated using the Euclidean and geodesic distance as well as 5 randomly generated LTMs. For each of these 7 topologies, an latent regression forest is trained on a subset of the training data and evaluated on sequence A.

Fig. 9 shows the standard evaluation metric of mean error, in mm, for each joint across the sequence. As shown, the Euclidean-generated LTM performs slightly better than the random ones, whereas the geodesic-generated LTM achieves the best performance on all joints except for two. In addition to this, we also employ more challenging metric, the proportion of test images that have *all* predicted joints within a certain maximum distance from the ground truth, which was recently proposed in [12]. The results using this metric can be seen in Fig. 10. As shown, the Euclidean-generated LTM achieves the same performance as the upper-bound of performance from the random LTMs, whereas the geodesic-generated LTM significantly outperforms all of them showing a 20% improvement at a threshold of $40mm$.

Additionally, we evaluate the impact of the cascaded error regressor. In Figure 11 we show the decrease in mean error distance for each joint across the whole sequence. As can be seen, we achieve up to a 22% reduction in mean error for one joint and and improvement of 10% on average.

In principle, since each tree generates much less votes comparing to traditional regression tree, more trees are needed in order to produce robust results. Figure 12 shows the accuracy impact from different number of trees. A reasonable choice considering the trade-off between accuracy and efficiency is 16 trees, which is the setting we use in all experiments.

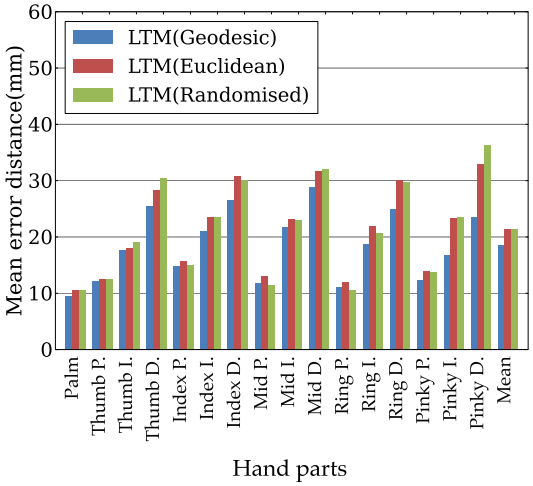


Fig. 9: Effect of different LTMs.

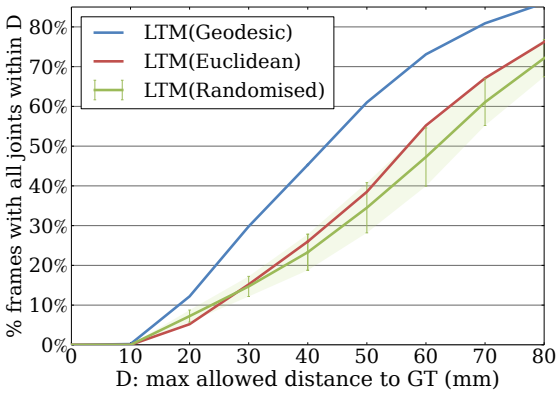


Fig. 10: Effect of different LTMs.

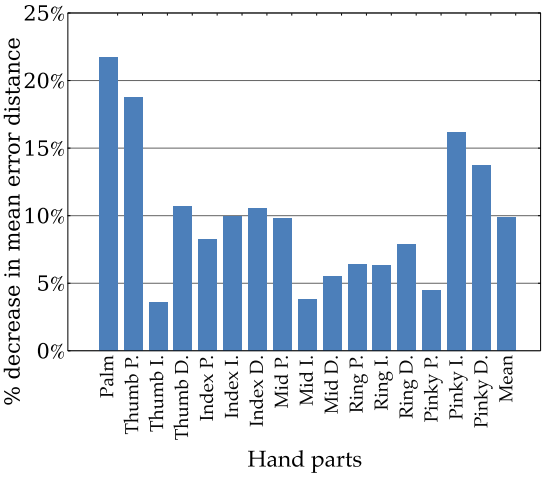


Fig. 11: Error reduction by error regression.

5.4 Comparison on the ICVL dataset

We compare LRF with two state-of-the-art methods. The first is a regression version of Keskin *et al* [14], for which we use our own implementation using the training parameters as described in [14]. The second method we compare to is the model-based tracker of Melax *et al* [3], for which we use a compiled binary version provided by the authors. As this method is model based it requires calibration of the hand structure (width and height). Therefore, in order to do a fair comparison we compare to two versions of this method, one which has been calibrated and one which has not.

In Fig. 13 (a) and (d) we show the cumulative moving average of the mean joint error. As can be seen, our approach maintains a low average error throughout both sequences, and as expected the tracking based approaches reduce in error over time. In Fig. 13 (b) and (e) we show the cumulative moving average of the palm prediction error, a stable part, whilst (c) and (f) show a relatively unstable part, index distal phalanx. Notice that after approximately the 500th frame the tracking based methods continuously decrease in accuracy for this joint, indicating the tracking has failed and could not recover. This further highlights the benefit of using frame-based approaches.

Additionally, in Fig. 14 and Fig. 15, we compare all methods using the more challenging metric proposed in [12]. As

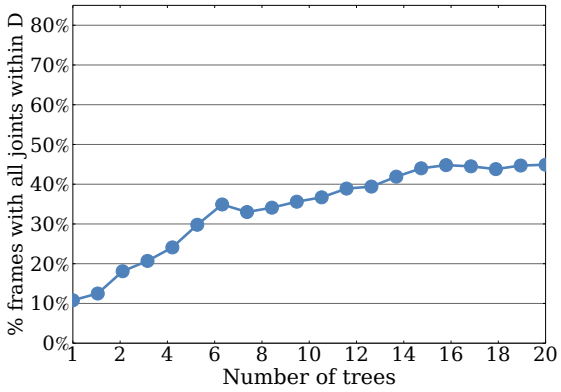


Fig. 12: Number of trees.

can be seen our method largely outperforms the other state-of-the-arts. An interesting observation is that the holistic baseline has better accuracy than the patch-based one. The reasons are two-fold: 1) The test images in this dataset are similar to training set. Hence the holistic baseline is less affected by its disadvantage, overfitting. 2) There is no kinematic constraint in the patch-based method. This leads to some rather displaced joint predictions, which brings down the accuracy, especially when using the criterion of worst case accuracy [12].

5.5 Comparison on the MSHD dataset

In this section we first compare LRF with the baselines on the MSHD dataset. To best visualise the trade-off between accuracy and efficiency, we plot the mean error against the time cost. Unlike the experiments with the ICVL dataset, the patch-based method has better accuracy than the holistic one, at the cost of much lower efficiency. This is because of the aforementioned sparse distribution of samples in this dataset, which requires more generalisation. However, LRF still outperforms both, and yet maintaining a fast speed.

Furthermore, we also compare with the method from Sharp *et al* [18] (kindly provided by the authors), which is a discriminative and generative hybrid method. Its generative part is a tracking-based method that built on PSO. In this section we only compare LRF with their discriminative part for two reasons: 1) LRF can be combined with PSO to be a hybrid method too. 2) The MSHD test set has no temporal coherence. In their experiment part, [18] perturbs the groundtruth as result from the previous frame to mimic tracking. In this case it is unclear how to compare LRF with the full pipeline fairly.

Similar to the patch-based baseline, the discriminative part of [18] is also forest-based and requires patches as input. However, rather than regressing on the full pose directly, it maps the input to certain pose distributions (close, open, fist, etc.) and draws samples from them. These samples are supposed to be optimised by PSO. For this experiment we only use their energy function to choose the best sample and measure its error. Usually the error reduces as more samples are drawn, but converges at some point. Thus the results appear to be a curve on the plot Fig. 17. Again, LRF can achieve better accuracy with lower time cost.

TABLE 1: Efficiency comparison.

Method	Time (fps)	Parallelism
LRF	62.5	None
Holistic	75	None
Patch-based [14]	8.6	None
Melax *et al* [3]	60	CPU (i7 2.2GHz)
Oikonomidis *et al* [27]	15	GPU (GTX 580)
Xu *et al* [45]	12	None
Sridhar *et al* [46]	10	GPU (NVS 300)
Qian *et al* [30]	25	CPU (i7 3.4GHz)

5.6 Efficiency

Regarding the run-time speed, our method runs in real-time at 62.5fps which is comparable to the holistic baseline (75fps) and much faster than the patch-based baseline [14] (8.6fps). Note that our method and the baselines are unoptimised—single threaded, without any CPU/GPU parallelism. In addition, we also report the speed of other methods such as Melax *et al* [3] (60fps), Oikonomidis *et al* [27] (15fps), Xu *et al* [45] (12fps), Sridhar *et al* [46] (10fps) and Qian *et al* [30] (25fps). Most of them are either CPU or GPU optimised. Despite that these results are reported with different hardwares, as indicated in Table 1, it is clear that LRF has the efficiency advantage.

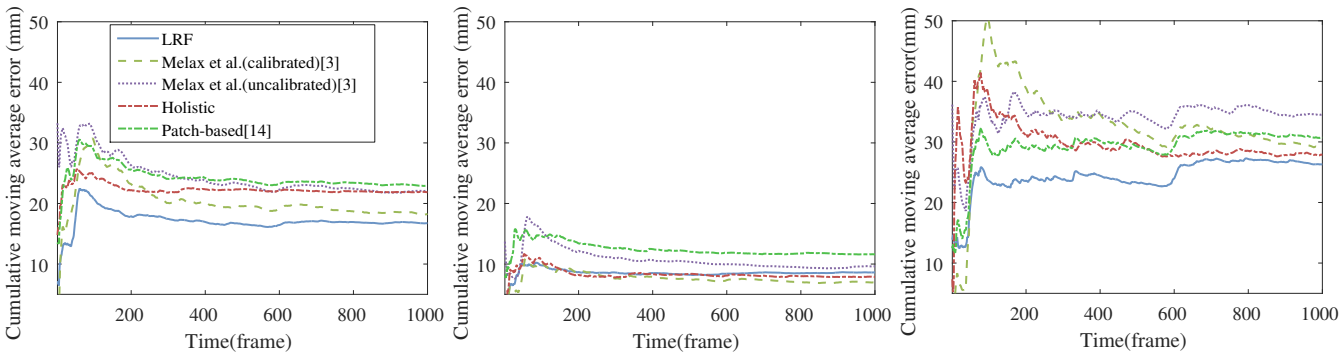
5.7 Qualitative Results

Qualitative results from the ICVL dataset are shown in Fig. 16. In the fail cases, despite that some of the part locations are wrong, especially distal phalanges which have larger depth in the LTM, the kinematic structure is preserved to some degree. However, due to lack of collision constraints, predictions of different parts are sometimes in the same location. In Fig. 18, examples from the MSHD dataset are shown. We order the result by mean error and visualise 5 images from the best, middle and worst cases respectively. The mid results tell us that errors usually appear at the distal phalanges. The worst results show that this method often struggles in the *egocentric* setting, where most of the joints are self-occluded. Finally a few shots from our demo video are shown in Fig. 19, which reflect the real-time performance of LRF. Due to the 30Hz frame-rate limit of the sensor, we cannot fully demonstrate the efficiency advantage in real scenario. Note that in the $7th$ frame, the hand moves out of the camera and moves back in the $8th$ frame. This is usually the point where a tracking-based method fails. However our method is single-frame based and will not be affected. For more details readers can refer to our demo video.[1]

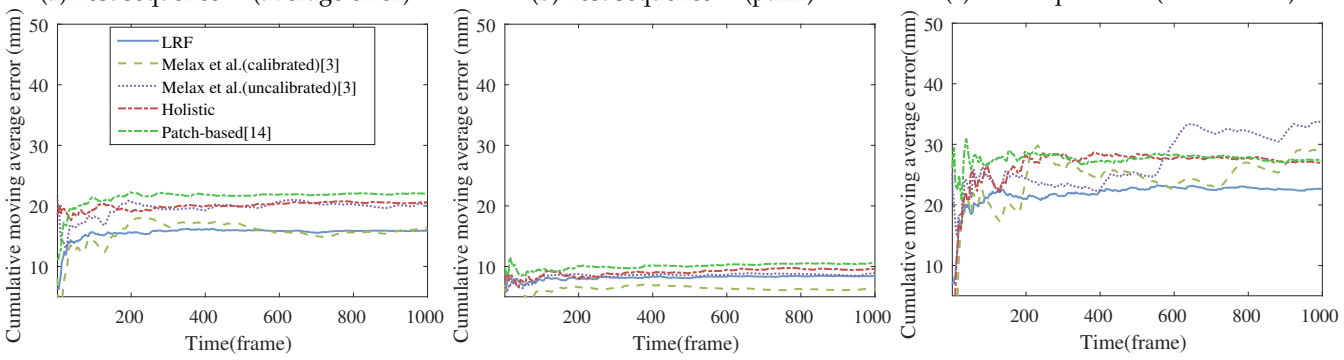
6 Discussion

In this paper we presented the latent regression forest, a method for real-time estimation of 3D articulated hand pose. Current discriminative methods in the field are either holistic or patch-based. Although patch-based methods achieve

1. https://youtu.be/-pluWWm60pM



(a) Test sequence A (average error) (b) Test sequence A (palm) (c) Test sequence A (index distal)



(d) Test sequence B(average error) (e) Test sequence B (palm) (f) Test sequence B (index distal)

Fig. 13: Quantitative comparison against state-of-the-art methods on ICVL dataset.

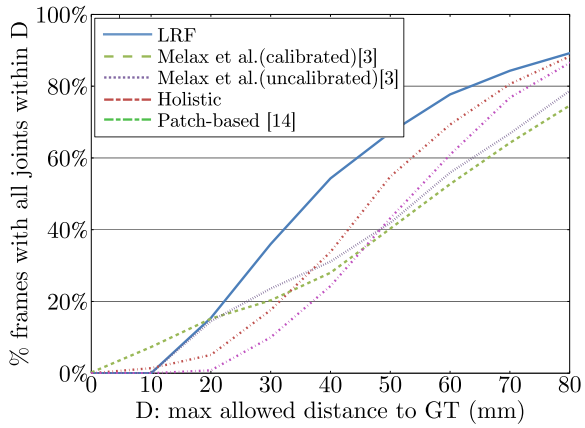


Fig. 14: Worst case accuracy on ICVL dataset (sequence A)

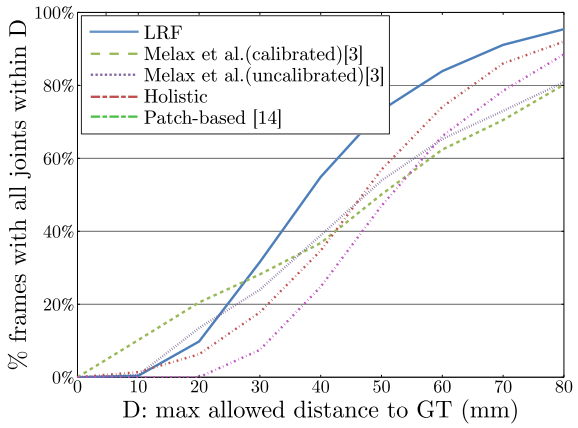


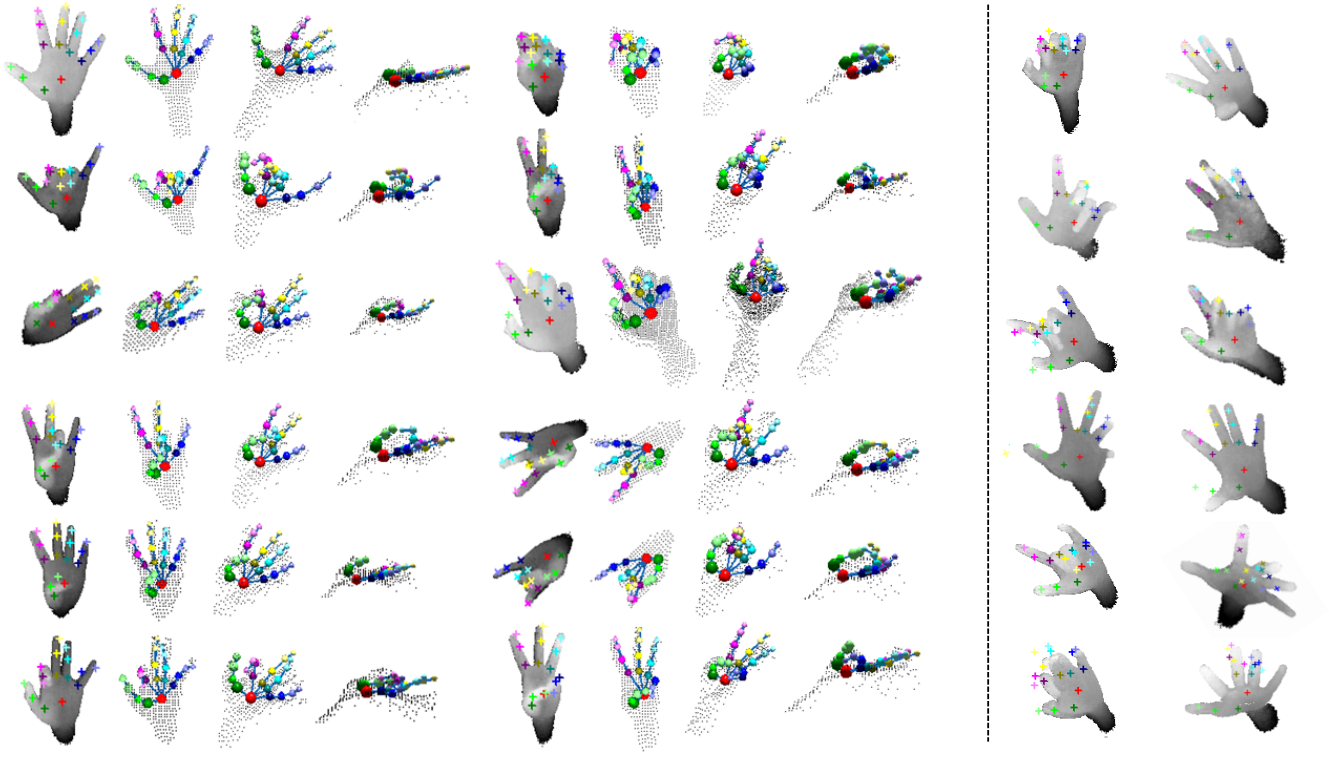
Fig. 15: Worst case accuracy on ICVL dataset (sequence B)

state-of-the-art performance, we manage to improve its efficiency by incorporating holistic concepts whilst keeping its generalisation power.

To the best of our knowledge this is the first work combining LTM and random forest. Since both have tree-based hierarchical structure, the integration is natural and easy to implement. This allows us to apply the LRF to many vision problems that need a structured search, either spatially or temporally. Moreover, LRF can potentially be applied to many existing topics of LTM from other fields, spanning from marketing to medicine. Readers may refer to [47] for more applications.

It is worth noting that we are not advocating discriminative methods are superior to generative ones. In the case of 3D hand pose estimation, the dimension of output space is too high to generate 'enough' training samples for fine-grained space coverage. Therefore a hybrid pipeline using a discriminative method as initialisation and a generative method as refinement is needed to support aforementioned applications. Readers can refer to [31] for a comparison between our method and such a pipeline on the Dexter dataset (results provided by us but trained on the ICVL dataset). What we are arguing is that our method improves both the accuracy and runtime efficiency of the discriminative part, hence will result in better performance of the full pipeline.

Besides the general limitations of a discriminative method, LRF also has a unique issue. Starting from holistic is efficient, but also makes LRF vulnerable for occlusions or inaccurate hand region segmentation. As demonstrated by the experiments, error at the first layer will accumulate in such a cascaded structure. To minimise this problem, a learning-based segmenter from [18] and occlusion handling



(a) Successes (b) Failures

Fig. 16: In (a) we show success cases; we show the localisation on the depth image followed by a visualisation of the estimated 3D joint locations from multiple angles. In (b) we show some failure cases, note however that the structure of the output is still in line with the hand topology. Image sizes have been normalised for visualisation.

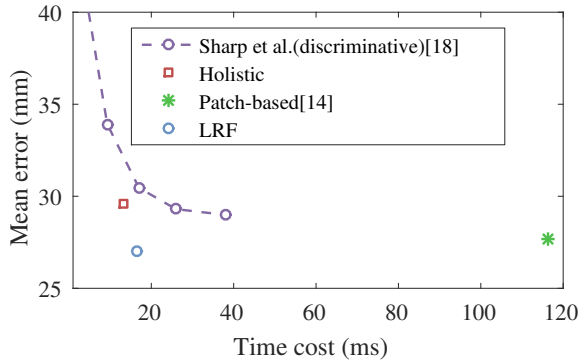


Fig. 17: Trade-off comparison on the MSHD dataset. Note that we only compare with the discriminative part of Sharp *et al* [18], which is CPU-optimised. Implementations of LRF and the other baselines do not use any CPU/GPU parallelism. Note that [18] predicts 21 joint locations. To be fair we do not consider the 5 finger tip positions when calculating the error of [18].

ideas from [48], [49], [50] can be combined.

Furthermore, although globally the results are constrained by the topology, locally it is still not guaranteed to be kinematically correct. Also the outputs are 3D locations, whilst an angular form is more desired as they follow anatomical structure and can be rendered. These two birds have been killed with one stone in [21], by straightforwardly using PSO to recover kinematically correct joint angles from a set of joint locations.

Acknowledgments

This project was supported by the Samsung Advanced Institute of Technology (SAIT). The authors would also like to thank Guillermo García-Hernando for the help of data collection and Mang Shao for the demo.

References

[1] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-Time Human Pose Recognition in Parts from Single Depth Images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2011.

[2] Y. Boussemart, F. Rioux, F. Rudzicz, M. Wozniewski, and J. R. Cooperstock, "A Framework for 3D Visualisation and Manipulation in an Immersive Space Using an Untethered Bimanual Gestural Interface," in *Proceedings of the ACM symposium on Virtual reality software and technology (VRST)*, 2004, pp. 162–165.

[3] S. Melax, L. Keselman, and S. Orsten, "Dynamics based 3D skeletal hand tracking," in *Proceedings of Graphics Interface*, 2013, pp. 63–70.

[4] Y. Jang, S.-T. Noh, H. J. Chang, T.-K. Kim, and W. Woo, "3D Finger CAPE: Clicking Action and Position Estimation under Self-Occlusions in Egocentric Viewpoint," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 21, no. 4, pp. 501–510, 2015.

[5] M. Hirsch, D. Lanman, H. Holtzman, and R. Raskar, "BiDi Screen: A Thin, Depth-Sensing LCD for 3D Interaction using Lights Fields," *ACM Transactions on Graphics (ToG)*, vol. 28, no. 5, December 2009.

[6] Y. Jacob, S. Manitsaris, F. Moutarde, G. Lele, and L. Pradere, "Hand gesture recognition for driver vehicle interaction," June 2015.

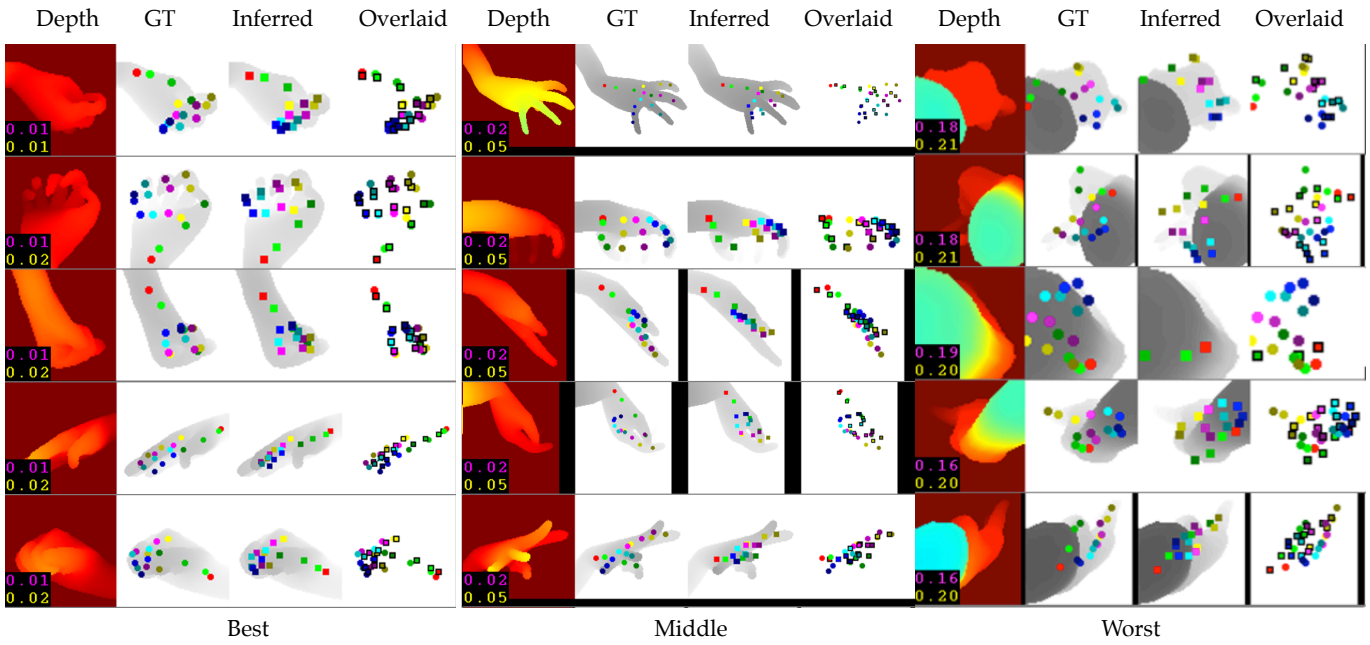


Fig. 18: **Qualitative results from the MSHD dataset in ascending order of mean error.** Depth image (in jet colour), groundtruth, inferred results, and inferred overlaid on groudtruth are shown. Numbers at bottom-left indicates the errors (pink: mean error, yellow: max error, unit: meter). Black stripes in some images are caused by the region of interest exceeding image edges.

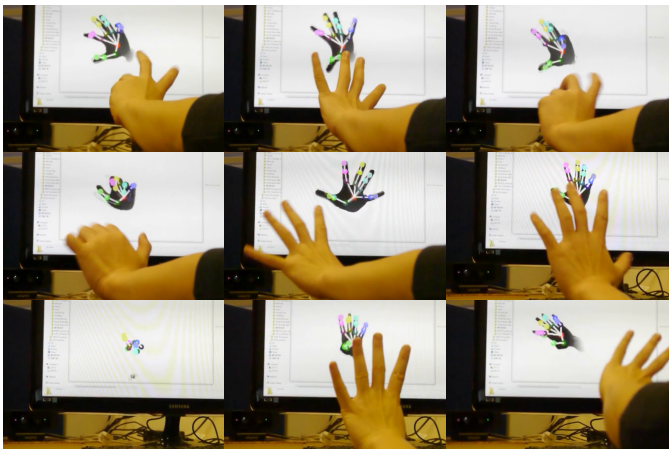


Fig. 19: Screen shots from the demo video that shows real-time performance. In frame #7 (bottom left), the target hand moves out of frame and moves back in later. However the accuracy is not affected in frame #8 and #9. For more details readers can refer to our demo video.

[7] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti, "American Sign Language Recognition with the Kinect," in *International Conference on Multimodal Interfaces (ICMI)*, 2011, pp. 279–286.

[8] H. J. Chang, G. Garcia-Hernando, D. Tang, and T.-K. Kim, "Spatio-Temporal Hough Forest for efficient detectionlocalisationrecognition of fingerwriting in egocentric camera," *Computer Vision and Image Understanding (CVIU)*, vol. 148, pp. 87–96, 2016.

[9] J. Romero, H. Kjellström, and D. Kragic, "Monocular real-time 3D articulated hand pose estimation," in *IEEE-RAS International Conference on Humanoid Robots*, December 2009, pp. 87–92.

[10] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 63:1–63:8, July 2009.

[11] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, "Efficient regression of general-activity human poses from depth images," in *IEEE International Conference on Computer Vision (ICCV)*, October 2011.

[12] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon, "The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012.

[13] G. Pons-Moll, J. Taylor, J. Shotton, A. Hertzmann, and A. Fitzgibbon, "Metric Regression Forests for Human Pose Estimation," in *Proceedings British Machine Vision Conference (BMVC)*, 2013, pp. 4.1–4.11.

[14] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun, "Hand pose estimation and hand shape classification using multi-layered randomized decision forests," in *European Conference on Computer Vision (ECCV)*, October 2012, pp. 852–863.

[15] D. Tang, T.-H. Yu, and T.-K. Kim, "Real-time Articulated Hand Pose Estimation using Semi-supervised Transductive Regression Forests," in *IEEE International Conference on Computer Vision (ICCV)*, December 2013, pp. 3224–3231.

[16] M. J. Choi, V. Y. F. Tan, A. Anandkumar, and A. S. Willsky, "Learning Latent Tree Graphical Models," *Journal of Machine Learning Research (JMLR)*, vol. 12, pp. 1771–1812, 2011.

[17] D. Tang, H.-J. Chang, A. Tejani, and T.-K. Kim, "Latent Regression Forest: Structured Estimation of 3D Hand Posture," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[18] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi, "Accurate, Robust, and Flexible Real-time Hand Tracking," in *Annual ACM Conference on Human Factors in Computing Systems (CHI)*, 2015, pp. 3633–3642.

[19] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding (CVIU)*, vol. 108, no. 1-2, pp. 52–73, 2007.

[20] J. S. Supancic III, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan, "Depth-based Hand Pose Estimation: Data, Methods, and Challenges," in *IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[21] J. Tompson, M. Stein, Y. LeCun, and K. Perlin, "Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks," *ACM Transactions on Graphics (ToG)*, vol. 33, no. 5, August 2014.

[22] P. Li, H. Ling, X. Li, and C. Liao, "3D Hand Pose Estimation Using Randomized Decision Forest with Segmentation Index Points," in

IEEE International Conference on Computer Vision (ICCV), 2015, pp. 819–827.

[23] J. Charles, T. Pfister, D. Magee, D. Hogg, and A. Zisserman, "Upper body pose estimation with temporal sequential forests," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2014.

[24] Y. J. Ho, S. Lee, S. H. Yong, and D. Y. Il, "Random Tree Walk Toward Instantaneous 3D Human Pose Estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 2467–2474.

[25] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded Hand Pose Regression," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 824–832.

[26] D. Tang, J. Taylor, P. Kholi, C. Keskin, T.-K. Kim, and J. Shotton, "Opening the black box: Hierarchical sampling optimization for estimating human hand pose," in *IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[27] N. K. Iason Oikonomidis and A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2011, pp. 101.1–101.11.

[28] M. de La Gorce, D. Fleet, and N. Paragios, "Model-Based 3D Hand Pose Estimation from Monocular Video," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 33, no. 9, pp. 1793–1805, 2011.

[29] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys, "Motion Capture of Hands in Action using Discriminative Salient Points," in *European Conference on Computer Vision (ECCV)*, October 2012, pp. 640–653.

[30] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Realtime and robust hand tracking from depth," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[31] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and Robust Hand Tracking Using Detection-Guided Optimization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[32] A. Tagliasacchi, M. Schroeder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, "Robust Articulated-ICP for Real-Time Hand Tracking," in *Eurographics Symposium on Geometry Processing (SGP)*, vol. 34, no. 5, 2015.

[33] G. Poier, K. Roditakis, S. Schulter, D. Michel, H. Bischof, and A. A. Argyros, "Hybrid One-Shot 3D Hand Pose Estimation by Exploiting Uncertainties," *CoRR*, 2015.

[34] P. Krejov, A. Gilbert, and R. Bowden, "Combining Discriminative and Model-based Approaches for Hand Pose Estimation," in *IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, May 2015.

[35] M. Oberweger, P. Wohlhart, and V. Lepetit, "Training a Feedback Loop for Hand Pose Estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3316–3324.

[36] C. Choi, A. Sinha, J. Hee Choi, S. Jang, and K. Ramani, "A Collaborative Filtering Approach to Real-Time Hand Pose Estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2336–2344.

[37] Y. Tian, C. Zitnick, and S. Narasimhan, "Exploring the Spatial Hierarchy of Mixture Models for Human Pose Estimation," in *European Conference on Computer Vision (ECCV)*, 2012, pp. 256–269.

[38] F. Wang and Y. Li, "Beyond Physical Connections: Tree Models in Human Pose Estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013, pp. 596–603.

[39] S. Harmeling and C. K. I. Williams, "Greedy Learning of Binary Latent Trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 33, no. 6, pp. 1087–1097, 2011.

[40] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, October 2001.

[41] A. Criminisi and J. Shotton, *Decision Forests for Computer Vision and Medical Image Analysis*. Springer Publishing Company, Incorporated, 2013.

[42] Y. Cheng, "Mean Shift, Mode Seeking, and Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 17, no. 8, pp. 790–799, 1995.

[43] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," in *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, September 2009, pp. 1393–1400.

[44] J. Taylor, R. Stebbing, V. Ramakrishna, C. Keskin, J. Shotton, S. Izadi, A. Hertzmann, and A. Fitzgibbon, "User-specific hand modeling from monocular depth sequences," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014, pp. 644–651.

[45] C. Xu and L. Cheng, "Efficient Hand Pose Estimation from a Single Depth Image," in *IEEE International Conference on Computer Vision (ICCV)*, December 2013, pp. 3456–3462.

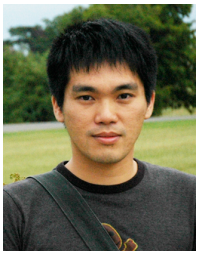
[46] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive Markerless Articulated Hand Motion Tracking using RGB and Depth Data," in *IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[47] R. Mourad, C. Sinoquet, N. L. Zhang, T. Liu, and P. Leray, "A Survey on Latent Tree Models and Applications," *Journal of Articial Intelligence Research (JAIR)*, vol. 47, pp. 157–203, 2013.

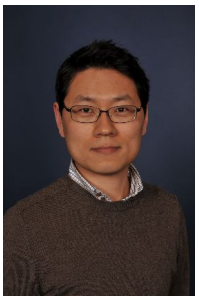
[48] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Full DOF Tracking of a Hand Interacting with an Object by Modeling Occlusions and Physical Constraints," in *IEEE International Conference on Computer Vision (ICCV)*, November 2011, pp. 2088–2095.

[49] D. Tzionas, A. Srikantha, P. Aponte, and J. Gall, "Capturing hand motion with an RGB-D sensor, fusing a generative model with salient points," *Pattern Recognition*, vol. 8753, pp. 277–289, 2014.

[50] D. Tzionas and J. Gall, "3D Object Reconstruction from Hand-Object Interactions," in *IEEE International Conference on Computer Vision (ICCV)*, December 2015.



Danhang Tang is a scientist and founding team member at perceptiveIO, Inc. Before that he was a visiting researcher at Microsoft Research Redmond. He obtained his PhD degree from Imperial College London. Prior to his PhD, he received a 1^{st} honour MSc degree from University College London and the BSc from Sun Yat-sen University. From 2007 to 2009, he worked as a system architect for Evryx Technologies Ltd., in support for SnapNow, one of the first image recognition apps in the world. During this time he co-drafted the visual search specification for China Telecom.



Hyung Jin Chang received his B.S. and Ph.D. degree from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Republic of Korea. He is a post doctoral researcher with the Department of Electrical and Electronic Engineering at Imperial College London. His current research interests include articulated structure learning, human robot interaction, object tracking, human action understanding and user modeling.

Alykhan Tejani received his MEng degree in Computing in 2010 and his MPhil whilst at the Computer Vision and Learning Lab in 2014, both at Imperial College London. He has also worked at Blippar where he was part of a team developing their visual search and augmented reality platforms. His current research interests include large-scale object recognition and 3D object pose estimation.

Tae-Kyun (T-K) Kim is an Assistant Professor and leader of Computer Vision and Learning Lab at Imperial College London, UK, since Nov 2010. He obtained his PhD from Univ. of Cambridge in 2008 and Junior Research Fellowship (governing body) of Sidney Sussex College, Univ. of Cambridge for 2007-2010. He has co-authored over 40 academic papers in top-tier conferences and journals in the field, his co-authored algorithm for face image retrieval is an international standard of MPEG-7 ISO/IEC. He is co-recipient of the KUKA best service robotics paper award at ICRA 2014, and general co-chair of CVPR15 workshop on HANDS and ICCV15 workshop on Object Pose.